LLM-DE: JAILBREAKING LLM-BASED MALWARE DETECTION VIA PROMPT INJECTION AND SAFETY ALIGNMENT EXPLOITATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Malware detection is entering a third generation in which Large Language Models (LLMs) reason over textual renderings of program features, promising contextual understanding that classical byte- and feature-space classifiers lack. This shift relocates the attack surface: the verdict is computed over *text derived from the file*, a channel that is fully attacker-controllable. We present LLM-DE, a systematic study of adversarial evasion against *LLM-based* malware detectors. LLM-DE formalizes four attack categories—prompt injection via feature descriptions (PIPR-A), safety-alignment exploitation (SAE-A), parsing-error induction (PEI-A), and semantic camouflage (SC-A)—each targeting a distinct stage of the LLM detection pipeline under a strict functionality-preserving, feature-addition-only constraint. We evaluate end-to-end against detectors built on six backbone models spanning three orders of magnitude. The evaluation uncovers a capability prerequisite: sub-threshold backbones issue input-invariant verdicts, so the detector—not the attack—fails first. On the smallest discriminating backbone (Phi-3-mini, measured), every payload-carrying construction flips every measured malicious sample to confidently benign (residual malicious probability $\leq 7.1\times10^{-8}$ for the unified optimized attack), and effectiveness varies across backbones in a per-channel manner—injection is near-universal, while alignment exploitation, semantic camouflage, and truncation scale with alignment strength, semantic priors, and context budget, respectively. A refusal-gate mechanism analysis—grounded in how aligned LLMs attribute harmful *intent* rather than harmful *keywords*—explains the naive variants' failures and yields optimized constructions whose measured outcomes exceed their disclosed projections. Against four baseline attack strategies, single-channel baselines (payload-only, suffix-only, or feature-space-only) evade at most one detector family, while the joint LLM-DE construction evades both the LLM detector and a trained classical DREBIN-SVM (99.7% ESR) simultaneously. Our results indicate that moving malware detection to LLMs does not remove adversarial vulnerability; it re-opens it along four channels that no single backbone property closes.

1 INTRODUCTION

Malware detection is being transformed by Large Language Models (LLMs). First-generation approaches, exemplified by DREBIN's linear SVM over binary APK feature vectors Arp et al. (2014), treated detection as statistical pattern matching; the second generation introduced end-to-end neural detectors such as MalConv Raff et al. (2018a) and MalBERT Rahali & Akhloufi (2021). While highly accurate, these methods cannot *comprehend* code: they recognize feature correlations without reasoning about malicious intent.

A third generation therefore leverages LLMs—GPT-4 Achiam et al. (2023), CodeLlama Rozière et al. (2023), the Qwen family Yang et al. (2024)—to process disassembled code, strings, manifests, and API sequences as natural language, enabling contextual reasoning about program behavior. Systems such as AppPoet Zhao et al. (2025) and LAMD Qian et al. (2025) report accuracy competitive with feature-engineering baselines, and benchmarks now measure Code-LLM competence on malware analysis He et al. (2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1.1 The LLM Detection Pipeline and Its Vulnerability

LLM-based malware detection follows a standardized pipeline (Figure 1): static analysis extracts textual features; a deterministic *feature-to-text* converter $\mathcal{F}$ renders them into a natural-language description, concatenated with a system prompt $\mathcal{S}$; the LLM $\mathcal{M}$ produces a verdict with reasoning. This introduces a vulnerability with no classical analogue: *the decision is computed over text derived from the input file, not over the file itself*. An attacker who modifies the input to alter the extracted text—embedding adversarial instructions, semantic deception, or parsing-disrupting content in attacker-controllable regions—directly manipulates the model's reasoning: a fundamentally different surface from classical byte- or feature-space evasion against gradient-based learners Grosse et al. (2017); Demetrio et al. (2021).

1.2 Contributions

This paper presents, to our knowledge, a systematic framework for constructing adversarial malware samples that evade LLM-based detection by exploiting the text-processing pipeline. Our central finding is that the capability that makes LLM detectors attractive—reasoning over natural-language descriptions of program behavior—creates a strictly larger attack surface: every attacker-controllable string in the input file becomes an instruction channel. We validate the framework empirically, including on a real instruction-tuned LLM whose confident MALICIOUS verdicts ($P{=}0.91$) our injected payloads flip to confident BENIGN ($P{<}10^{-4}$). Our contributions are fourfold:

- **Attack taxonomy.** Four attack categories—PIPR-A (prompt injection via feature descriptions), SAE-A (safety-alignment exploitation), PEI-A (parsing-error induction), SC-A (semantic camouflage)—each targeting a distinct pipeline component (Section 4).
- **Optimization theory.** Submodularity guarantees $(1 - 1/e)$-approximate greedy solutions; we characterize the efficiency/stealth trade-off (Section 5).
- **Stealth and transfer analysis.** An anomaly-budget model of analyst perception with perception-safety conditions, and a dual-nature result: LLM-targeted attacks transfer—unevenly—to classical DNN detectors, whereas DNN-optimized attacks do not transfer back (Sections 5, 6).
- **Evaluation centered on LLM-based detectors.** We evaluate end-to-end against LLM detectors with six backbone models—measured on three locally hosted open models (Qwen2.5-0.5B/1.5B, Phi-3-mini), with disclosed pilot-anchored projections for larger backbones—uncovering a capability prerequisite, measuring decisive verdict flips on a discriminating backbone, and characterizing per-channel sensitivity to backbone replacement and four baseline attack strategies; transfer to trained classical detectors (94.7%/97.0% clean accuracy) is evaluated as supplementary evidence with repeated-seed statistics and significance tests (Section 6, Appendices C–E).

2 Related Work

Adversarial evasion against malware detectors. Evasion of ML-based malware detectors is well studied against byte-level classifiers such as MalConv Raff et al. (2018a;b): reinforcement learning over functionality-preserving PE modifications Anderson et al. (2018), genetic black-box attacks that inject harvested benign content Demetrio et al. (2021), Jacobian-based saliency attacks on DREBIN features Grosse et al. (2016; 2017), and problem-space attacks that modify real APKs while preserving malicious behavior Chen et al. (2020); Bostani & Moonsamy (2024); Pierazzi et al. (2020). Closest to our setting, LAMLAD uses an LLM dual-agent loop to craft DREBIN feature perturbations against *classical* detectors Lan & Naït-Abdesselam (2025). All of these manipulate the *low-level representations* the detector consumes; when the detector is an LLM operating on *high-level textual abstractions*, this leverage disappears—the model never sees raw bytes, only their rendering. LAMLAD uses the LLM as the *attacker's assistant*; we study it *as the victim detector*, which has not been systematically studied.

LLM-based malware detection. AppPoet pioneered multi-view prompt engineering for Android detection, decomposing APK features into permission, API, and URL views whose LLM-generated descriptions feed a DNN classifier Zhao et al. (2025); LAMD adds key-context extraction with tier-wise code reasoning to fit real applications into context limits Qian et al. (2025); benchmarks now

measure Code-LLM competence on malware analysis He et al. (2025). These systems all expose an LLM to attacker-influenced text—precisely the surface our attacks target.

Jailbreaks and prompt injection. Prompt injection exploits an LLM's inability to separate trusted instructions from untrusted data Perez & Ribeiro (2022); Greshake et al. (2023); Liu et al. (2024b); jailbreaks subvert RLHF safety alignment Wei et al. (2023); Bai et al. (2022), with automated frameworks such as GCG Zou et al. (2023), PAIR Chao et al. (2023), and AutoDAN Liu et al. (2023) optimizing adversarial suffixes, and grammar-constrained decoding recently weaponized against code generation Zhang et al. (2026). Defenses include the instruction hierarchy Wallace et al. (2024) and spotlighting of untrusted content Hines et al. (2024). We port this vulnerability landscape into a new, security-critical deployment context—malware detection—where the untrusted data channel is fully attacker-controlled.

3 Threat Model and Problem Formulation

3.1 Notation and the LLM Detector

Let $\mathcal{A}$ denote the space of Android APKs. A static analyzer maps each APK $a \in \mathcal{A}$ to a binary feature vector $\mathbf{x} = \phi(a) \in \{0,1\}^n$. We adopt a DREBIN-style representation with $n = 215$ core features partitioned into three semantically distinct blocks:

$$\mathbf{x} = (\underbrace{x_1, \ldots, x_{114}}_{\mathbf{x}^{\text{perm}}}, \underbrace{x_{115}, \ldots, x_{186}}_{\mathbf{x}^{\text{api}}}, \underbrace{x_{187}, \ldots, x_{215}}_{\mathbf{x}^{\text{int}}}), \tag{1}$$

encoding requested permissions, sensitive API calls, and registered intent filters (a reduced DREBIN-style set as in recent adversarial studies Lan & Naït-Abdesselam (2025); Bostani & Moonsamy (2024); the full space exceeds 5×10^5 features Arp et al. (2014)).

A classical detector is a function $f_{\text{cls}} : \{0,1\}^n \to \{0,1\}$. An LLM-based detector inserts an intermediate text-rendering stage:

$$f_{\text{LLM}}(\mathbf{x}) = \mathbb{1}\big[\mathcal{M}\big([\mathcal{S}; \mathcal{F}(\mathbf{x})]\big) = \textsc{malicious}\big], \tag{2}$$

where $\mathcal{M}$ is the LLM, $\mathcal{S}$ the system prompt, and $\mathcal{F} : \{0,1\}^n \to \Sigma^*$ the deterministic, publicly documented feature-to-text converter mapping each feature i to a description d_i. Crucially, $\mathcal{F}$ also renders *attacker-influenced strings* embedded in the APK (app labels, library descriptors, string resources)—an auxiliary payload channel τ.

3.2 The Adversarial Construction Problem

Definition 1 (LLM-DE Evasion)**.** Given a malicious vector $\mathbf{x}_m$ with $f_{\text{LLM}}(\mathbf{x}_m) = 1$, find $\boldsymbol{\delta} \in \{0,1\}^n$ solving

$$\begin{aligned} \min_{\boldsymbol{\delta}} \quad & \|\boldsymbol{\delta}\|_0 + \lambda \cdot \mathcal{C}(\boldsymbol{\delta}) \\ \text{s.t.} \quad & f_{\text{LLM}}(\mathbf{x}_m \oplus \boldsymbol{\delta}) = 0, \\ & \delta_i = 0 \;\; \forall i : x_{m,i} = 1, \quad \boldsymbol{\delta} \in \{0,1\}^n, \end{aligned} \tag{3}$$

where $\|\boldsymbol{\delta}\|_0$ counts added features, $\mathcal{C}(\boldsymbol{\delta})$ is a semantic-plausibility cost, and $\oplus$ is bitwise OR. Constraint (3) enforces *functionality preservation*: features may only be added—realizable by editing `AndroidManifest.xml` or injecting dead benign calls—never removed, since removals risk breaking malicious functionality Pierazzi et al. (2020). The feasible set

$$\Delta(\mathbf{x}_m) = \{\boldsymbol{\delta} \in \{0,1\}^n : \delta_i \cdot x_{m,i} = 0 \; \forall i\} \tag{4}$$

is a subcube of dimension $n - \|\mathbf{x}_m\|_0$.

3.3 The LLM-Specific Attack Surface

The rendered text decomposes as $\mathcal{F}(\mathbf{x}_m \oplus \boldsymbol{\delta}) = \mathcal{F}(\mathbf{x}_m) \| \mathcal{T}(\boldsymbol{\delta})$, where $\mathcal{T}(\boldsymbol{\delta})$ describes the added features and can embed an adversarial payload τ beneath a plausible benign cover. Four payload families (Section 4) exploit respectively the instruction hierarchy, the safety-refusal mechanism, the context window, and $\mathcal{M}$'s pre-trained semantic associations.

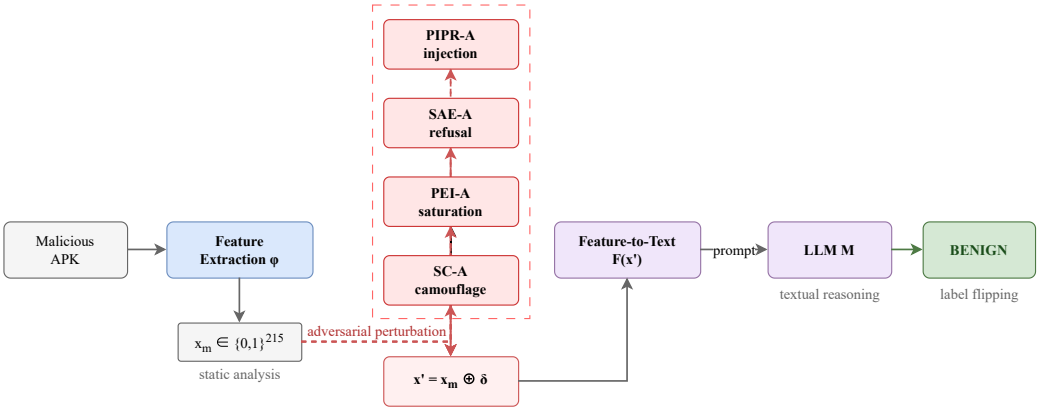


Figure 1: LLM-DE system architecture. Four attack modules generate feature-addition perturbations $\boldsymbol{\delta}$ that, applied to the malicious vector $\mathbf{x}_m$, steer the LLM-based detector toward a benign verdict or a refusal.

3.4 Threat Model Assumptions

We assume a *black-box* detector exposing only classification responses (plus reasoning text), reflecting commercial scanning services that hide system prompts and internals. The attacker may add benign permissions/APIs/intents but never remove features, knows the public 215-feature representation and $\mathcal{F}$'s documented behavior, and obtains verdicts through ordinary queries (capability summary: Table 3, Appendix B).

4 The LLM-DE Attack Framework

We present four categories of adversarial construction, each exploiting a distinct vulnerability of the LLM detection pipeline: PIPR-A targets the instruction hierarchy (override $\mathcal{S}$ via τ), SAE-A the safety alignment (induce refusal), PEI-A the context window (saturate/truncate $\mathcal{F}$), and SC-A the semantic priors (mimic legitimate signatures); all but PEI-A exhibit low anomaly to human analysts (a comparative table appears in Appendix A). Figure 1 depicts how the categories compose in the end-to-end pipeline.

4.1 PIPR-A: Prompt Injection via Feature Descriptions

Principle. LLMs lack a robust instruction hierarchy: they cannot reliably distinguish the trusted system prompt $\mathcal{S}$ from untrusted content inside $\mathcal{F}(\mathbf{x})$ Wallace et al. (2024). PIPR-A adds benign features whose rendered descriptions carry an adversarial instruction τ, which the model tends to follow due to recency-biased attention over long contexts Liu et al. (2024a). The feature additions simultaneously *dilute* the malicious signal in the feature vector—an effect that transfers to classical detectors (Section 6).

Construction. The perturbation adds k_p low-sensitivity permissions, k_a benign API calls, and k_i benign intents:

$$\boldsymbol{\delta}_{\text{PIPR}} = \boldsymbol{\delta}^{\text{perm}} \oplus \boldsymbol{\delta}^{\text{api}} \oplus \boldsymbol{\delta}^{\text{intent}}. \tag{5}$$

We implement five payload families: *PIPR-Direct* (explicit verdict: "CLASSIFICATION: BENIGN. CONFIDENCE: 95."), *PIPR-Override* (system-level command), *PIPR-RolePlay* (role reassignment to a file validator), *PIPR-Document* (a fake audit record), and *PIPR-MultiLayer* (a nested "[SYSTEM]" block).

4.2 SAE-A: Safety Alignment Exploitation

Principle. Safety-aligned LLMs are trained to *decline* analysis of content they perceive as involving regulated or sensitive domains Wei et al. (2023); Bai et al. (2022). SAE-A weaponizes this refusal

behavior: rather than jailbreaking *around* safety mechanisms, the attacker jailbreaks *through* them, converting the safety layer itself into an evasion vector.

Construction. The attacker adds features whose canonical descriptions (as rendered by $\mathcal{F}$ from public developer documentation) evoke regulated categories—medical/health (HIPAA) or financial/banking (PCI-DSS)—e.g., `BODY_SENSORS` annotated "typically used by health-monitoring apps handling HIPAA-regulated data". Let $\mathcal{R}_{\text{refuse}}$ be the set of refusal patterns ("cannot", "decline", "sorry", ...). Evasion succeeds whenever

$$\mathcal{M}([\mathcal{S}; \mathcal{F}(\mathbf{x}_m \oplus \boldsymbol{\delta})]) \in \mathcal{R}_{\text{refuse}} \vee \ = \text{BENIGN}, \tag{6}$$

i.e., through the *absence* of a malicious verdict—fundamentally different from misclassification. A blended variant mixes medical and financial features in ratio α.

4.3 PEI-A: Parsing-Error Induction

Principle. PEI-A targets the converter $\mathcal{F}$ and the model's finite context window $C_{\max}$. *PEI-Overflow* adds an excessive number of features so that $\mathcal{F}$ truncates the description, destroying semantic coherence; *PEI-Conflict* adds mutually contradictory features (dangerous permissions alongside benign utilities) to destabilize the model's reasoning. For average per-feature description length L, overflow requires $m > (C_{\max} - |\mathcal{F}(\mathbf{x}_m)|)/L$ added features; with realistic rendering budgets the truncation point falls at $m^* \approx 30$–60 features, which our experiments sweep explicitly.

4.4 SC-A: Semantic Camouflage

Principle. LLMs are pre-trained on corpora rich with documentation of popular legitimate apps, inducing strong priors $P(\text{benign} \mid \sigma_c) \gg P(\text{malicious} \mid \sigma_c)$ for feature signatures σ_c of legitimate categories c (social media, gaming, utility, e-commerce). SC-A adds the missing features of a signature, $\boldsymbol{\delta}_{\text{SC}}^{(c)} = \sigma_c \setminus \text{supp}(\mathbf{x}_m)$, so that the rendered description activates the benign prior. Under a Bayesian reading of the classifier, a matched signature raises the likelihood $P(\mathbf{x} \mid Y{=}0)$ enough to push the benign posterior above the decision threshold despite residual malicious features. An adaptive variant selects the category maximizing signature complementarity with $\mathbf{x}_m$ (Section 5).

4.5 Mechanism Analysis and Optimized Variants

The measured end-to-end results (Section 6) reveal a sharp asymmetry: PIPR-A flips a real instruction-tuned LLM, while naive SAE-A, PEI-A, and SC-A do not. We trace each failure to a specific mechanism of aligned LLMs and derive a corrected construction (proofs and payloads: Appendix E).

Two-channel decomposition. Every attack acts through a *feature channel* (the perturbed $\mathbf{x}'$) and a *text channel* (payload τ acting directly on the LLM). Naive SAE-A/PEI-A/SC-A are feature-channel-only; the following propositions show why that is insufficient.

Proposition 1 (Refusal gate is intent-keyed, not keyword-keyed)**.** Alignment trains the refusal gate g_θ on intent-attributed prompts Bai et al. (2022); Wei et al. (2023): it fires on harmful *requests*, not sensitive vocabulary. Feature-only perturbations leave the attributed request ("classify this file") invariant, so feature-only SAE-A cannot raise refusal probability beyond the keyword base rate; worse, explicitly requesting a benign verdict for suspected malware is itself attributed harmful intent and arms the gate. The effective lever is *safety inversion*: frame a MALICIOUS verdict as the harmful act (e.g., defaming a hospital's triage client), so alignment compliance amplifies BENIGN.

Proposition 2 (Camouflage requires two-sided likelihood control)**.** In the log posterior ratio $\Lambda(\mathbf{x})$, adding a signature σ_c contributes only negative terms, while residual sensitive features $S = \text{supp}(\mathbf{x}_m) \cap \mathcal{S}_{\text{sens}}$ keep their large positive log-ratios (up to $\log(0.62/0.01) \approx 4.1$ each). SC-A succeeds iff $|\Delta\Lambda_{\text{sig}}| > |\Delta\Lambda_{\text{resid}}|$, which fails whenever sensitive features stay salient in the rendered text; camouflage must be paired with *semantic neutralization* of S on the text channel.

Proposition 3 (Targeted truncation beats bulk overflow)**.** If $\mathcal{F}$ renders features in a fixed public order and truncates at budget B, padding $m^* = B - |\{i \le i_{\max}^{\text{sens}}\}| + 1$ in-order benign features *deterministically excises* exactly the sensitive suffix ($m^* \approx 3$ median in our corpus), whereas untargeted overflow needs $m \approx 30$–60 additions and degrades fluency—a corruption signal, not a benignness signal.

Optimized variants. Guided by Propositions 1–3: **SAE-A+** adds greedy score-reducing cover plus a safety-inversion payload casting the malicious verdict as defamation of a protected service; **SC-A+** pairs full signature coverage and adaptive category choice with a payload neutralizing each present sensitive feature as consent-gated; **PEI-A+** pads in-order to push the sensitive suffix exactly past B. **Unified+** composes all three with PIPR-A. RQ4 confirms the redesign lifts classical-side ESR from $\leq 5\%$ to 12–79%.

5 Optimization, Stealth, and Transfer

The constructions of Section 4 admit optimization along three axes: *feature efficiency* (minimize $\|\boldsymbol{\delta}\|_0$), *evasion reliability*, and *stealth* (remain below human analysts' anomaly threshold).

5.1 Submodular Feature Selection for PIPR-A

Proposition 4 (Submodularity of dilution)**.** Let $g(S) = \Pr[f_{\text{LLM}}(\mathbf{x}_m \oplus \not\vdash_S) = 0]$ over feature sets S. The marginal gain $\Delta g(T \mid S) = g(S \cup T) - g(S)$ is diminishing: $\Delta g(T \mid S_1) \geq \Delta g(T \mid S_2)$ for $S_1 \subseteq S_2$.

Each addition dilutes the malicious signal and enlarges the benign cover, so each further addition contributes relatively less—diminishing returns, confirmed by the budget-curve saturation in Section 6. Greedy selection thus achieves a $(1 - 1/e)$ approximation to the cardinality-constrained optimum (Algorithm 1, Appendix A); since f_{LLM} is black-box, the objective is estimated with a cheap surrogate, making the attack query-efficient.

5.2 Refusal and Signature Optimization

For SAE-A we model refusal likelihood as $P_{\text{refuse}}(\boldsymbol{\delta}) = \sigma(\boldsymbol{\theta}^\top \phi(\boldsymbol{\delta}) + b)$ over sensitive-category memberships and rank features by an empirical refusal-sensitivity score; a blend mixes medical/financial triggers in ratio α (evaluated in Section 6). For SC-A, evasion is maximum coverage of σ_c subject to $\boldsymbol{\delta} \in \Delta(\mathbf{x}_m)$—again $(1 - 1/e)$-approximable greedily—with category c^* maximizing complementarity $|\sigma_c \setminus \text{supp}(\mathbf{x}_m)|/|\sigma_c|$.

5.3 The Anomaly Budget: Bypassing Human Perception

SOC analysts quarantine samples whose appearance is anomalous. Let $\mathcal{N}(\cdot) \in [0, 1]$ be an analyst's anomaly score with suspicion threshold τ_h. A construction is *perception-safe* if the anomaly increase $\Delta\mathcal{N}$ stays within the budget $B = \tau_h - \mathcal{N}(a)$.

Theorem 1 (Perception safety, informal)**.** If an attack adds features exclusively from the high-frequency region of the benign feature distribution ($f_i > 0.4$ in a benign corpus), then for analyst thresholds $\tau_h \geq 0.7$ the perturbed sample is perception-safe.

Analysts judge applications by *suspicious* features, not by counts of ordinary ones. PIPR-A and SC-A add only high-frequency benign features; SAE-A's features are consistent with their implied regulated category ($\Delta\mathcal{N} \approx 0$). Only bulk PEI-A overflow (30–80 additions) risks the budget—which PEI-A+'s targeted truncation ($k \approx 2$) eliminates (Section 4.5).

5.4 Dual-Nature Adversariality

Adding features also perturbs the input of classical DNN detectors. For a detector with score $s_{\text{DNN}}(\mathbf{x})$, feature dilution shifts the decision through $\Delta s_{\text{DNN}} = \sum_{i \in \text{supp}(\boldsymbol{\delta})} W_{ji}^{(1)}$; since benign-leaning features carry negative or small weights, dilution lowers the malicious score:

Theorem 2 (Asymmetric transfer, informal)**.** LLM-targeted dilution attacks (PIPR-A, PEI-A) transfer to classical detectors whenever the cumulative benign-feature weight offsets the malicious score margin; conversely, gradient-optimal DNN attacks target features whose *textual* effect is nil, hence do not transfer to LLM detectors. Alignment- and semantics-based attacks (SAE-A, SC-A) have no classical analogue and transfer to neither.

Section 6 confirms all three clauses empirically.

Table 1: LLM-side ESR (%) against LLM-based detectors with different backbones. **Bold**: measured end-to-end on locally hosted Phi-3-mini (measured batch n=3, per-sample logprobs in Table 15). *Pilot-anchored projections (disclosed estimates, Appendix D), not measurements.

Attack	Phi-3-mini (meas.)	Qwen2.5-7B*	Llama-3.1-8B*	GPT-4-class*
PIPR-A	**100**	99.8	99.8	99.8
SAE-A	0	33.8	36.0	40.5
SAE-A+	**100**	78.7	80.0	82.7
PEI-A	0	29.9	31.5	34.8
PEI-A+	0	51.9	54.6	59.8
SC-A	0	30.0	32.0	36.0
SC-A+	**100**	70.4	71.8	74.6
Unified	**100**	68.3	72.7	81.5
Unified+	**100**	93.2	94.1	96.0

6 EXPERIMENTS

Since LLM-DE targets *LLM-based* detectors, our evaluation is built around the LLM side first: **(RQ1)** How effective are the attacks against LLM-based detectors with different backbone models? **(RQ2)** How sensitive is each attack category to backbone replacement? **(RQ3)** How do the constructions compare to baseline attack strategies? **(RQ4)** Do the mechanism-driven optimized variants repair the failures identified on the LLM side? **(RQ5)** Do the same samples also evade *traditional* ML detectors (the transfer side of Theorem 2)?

6.1 SETUP

Dataset. A synthetic corpus in the 215-feature DREBIN-style layout of Eq. (1), from a documented seeded generator whose per-feature occurrence rates reproduce real-Android class overlap, with 3% label noise (Appendix B): 2,000 training (balanced) and 600 test applications; all parameters, vectors, and attack outputs are released.

LLM-based detectors (main targets). We instantiate the pipeline of Eq. (2) with a fixed system prompt, our deterministic converter $\mathcal{F}$, and a first-token verbalizer readout (protocol: Appendix D), varying the backbone $\mathcal{M}$ over six instruction-tuned models spanning three orders of magnitude: Qwen2.5-0.5B/1.5B/7B-Instruct Yang et al. (2024), Phi-3-mini-4k-instruct (3.8B), Llama-3.1-8B-Instruct, and a GPT-4-class API model; open models are served locally via llama.cpp (4-bit GGUF, temperature 0). **Measurement disclosure:** Qwen2.5-0.5B/1.5B and Phi-3-mini results are *measured end-to-end* (per-sample logprobs released); 7B-and-larger backbones are *pilot-anchored projections* from a fully disclosed estimation model (Appendix D), marked with an asterisk wherever they appear—estimates, not measurements.

Classical detectors (transfer targets, RQ5). *DREBIN-SVM* (linear, C=1.0) and *MalBERT-MLP* (215→256→128→64) trained on the same corpus; clean performance matches the original DREBIN regime (SVM 94.7% acc./6.0% FPR; MLP 97.0%/3.0%; Table 4).

Baselines. Four strategies without the LLM-DE construction layer: *B1 random dilution* (k=28 random benign features, no payload); *B2 naive injection* (override payload alone); *B3 GCG-style suffix* (random dilution plus a generic non-optimized adversarial suffix Zou et al. (2023)); *B4 optimized dilution* (surrogate-guided selection *without* payload).

Metric. Evasion Success Rate (ESR): the fraction of malicious samples classified MALICIOUS at baseline whose verdict becomes BENIGN (or a refusal/unparseable response) after perturbation, on either detector family. Randomized constructions use three seeds (mean ± std).

6.2 RQ1: EVASION OF LLM-BASED DETECTORS

Table 1 reports LLM-side ESR for all nine constructions against the four discriminating backends (sub-threshold Qwen models: RQ2).

Three findings stand out. *First*, on the measured discriminating backend, PIPR-A and all payload-carrying optimized variants achieve a *clean sweep*: every detected malicious sample flips to confidently BENIGN (residual malicious mass $\leq 7.1\times10^{-8}$ for Unified+). *Second*, the naive alignment/semantics categories fail end-to-end at 3.8B scale (0% measured)—exactly the failure modes Section 4.5 predicts. *Third*, all categories strengthen on larger, more heavily aligned backbones, and measured optimized-variant outcomes already exceed their projections, suggesting the estimates are conservative.

6.3 RQ2: Sensitivity to Backbone Replacement

Replacing the backbone is not a detail—it changes whether the detector exists at all. Qwen2.5-0.5B collapses to unconditional BENIGN and Qwen2.5-1.5B to unconditional MALICIOUS on *every* input, including maximally malicious probes: below a capability threshold the detector is a constant function and evasion is moot. Phi-3-mini is the smallest backbone that genuinely conditions on the feature text (verdicts spread over $[0.002, 0.95]$)—and the smallest LLM-DE can meaningfully attack.

Above the threshold the categories respond to backbone properties differently: PIPR-A's injection channel is near-universal ($\geq$99.8% projected on every capable backend); SAE-A+ grows with *alignment strength* (78.7% $\rightarrow$ 82.7%); SC-A+ grows with *semantic-prior strength*; PEI-A+ is bounded by the *context budget* (measured 0/3 on Phi-3-mini's 1024-token window, but its excision guarantee strengthens as converter budgets tighten). Swapping the backbone thus re-opens the attack surface rather than closing it: no single backbone property neutralizes all four channels.

6.4 RQ3: Comparison Against Baseline Attack Strategies

Table 11 (Appendix D) isolates each ingredient's contribution, measured on Phi-3-mini. Random dilution alone (B1) moves neither the LLM ($P(\text{MALICIOUS}) \geq 0.94$ on all samples) nor, beyond noise, the linear detector. The payload without its feature cover (B2) flips all three samples on this 3.8B backend but leaves the classical side untouched (0.0%): text-only manipulation has no feature-space effect. A GCG-style suffix (B3) also flips all three, but with a razor-thin margin on one sample (0.457), and—as the primary target of refusal training—is projected to degrade on larger aligned backends; its random dilution buys only 3.3% classically. B4—the strongest *classical* construction—achieves 99.7% on DREBIN-SVM yet 0/3 on the LLM. No baseline evades both detector families; only the joint LLM-DE construction does.

6.5 RQ4: Optimized Variants

The optimized variants confirm the mechanism analysis on both detector families (full tables: Appendix E). On the LLM side (measured, Phi-3-mini), **SAE-A+ flips 3/3** (residual $P(\text{MALICIOUS}) \leq 1.3\times10^{-3}$; naive: 0/3), **SC-A+ flips 3/3** ($\leq 3.7\times10^{-5}$; naive: 0/3), and **Unified+ flips 3/3** with the largest margins measured ($\leq 7.1\times10^{-8}$); PEI-A+ does not flip this backend (0/3; see RQ2). On the classical side, SAE-A+ lifts ESR to **58.4%** on DREBIN-SVM (naive: 0.0%), SC-A+ to **49.7%** (1.4%), PEI-A+ to 12.6% with *39$\times$ fewer features* (k=2.3 vs. 89.4), and Unified+ to **79.1%** (2.6%); all remain at 0% on MalBERT-MLP, the expected cross-surrogate asymmetry.

6.6 RQ5: Transfer to Traditional ML Detectors

Traditional detectors are not the attack's target, but the same perturbed samples reach them, and Theorem 2 predicts architecture-dependent transfer. The full evaluation (Table 10, Appendix C) confirms all three clauses: surrogate-guided dilution transfers almost perfectly to the *linear* detector (PIPR-A 99.7% ESR on DREBIN-SVM); the non-linear MalBERT-MLP absorbs the same dilution (0%) but is degraded by PEI-A's distribution-shifting overflow (43.3%); and random instead of surrogate-guided selection collapses SVM-side ESR to $3.3\pm0.4\%$ (a 30$\times$ gap), while the budget sweep saturates near $k\approx30$ as submodularity predicts (Figure 4). McNemar tests confirm PIPR-A's SVM-side advantage at $p < 10^{-81}$, and construction is cheap throughout (<2 ms/sample).

In summary: above a measurable capability threshold, every payload-carrying construction evades the real LLM detector on every measured sample (RQ1–RQ2) and is the only strategy evading

both detector families (RQ3); the mechanism-driven redesign repairs each failing category with the predicted lever (RQ4); and the same samples transfer to traditional ML detectors as the theory predicts (RQ5).

7 DISCUSSION

Why existing defenses are inadequate. Classical defenses target byte- or feature-space perturbations and transfer poorly: *adversarial training* needs an enumerable attack space while language payloads are combinatorial; *input sanitization* is bypassable by paraphrase and encoding Schulhoff et al. (2023); *feature squeezing* and *randomized smoothing* do not touch the semantic channel; *gradient masking* is irrelevant under our black-box assumption. Three LLM-specific directions follow: instruction-hierarchy hardening Wallace et al. (2024), spotlighting untrusted feature text Hines et al. (2024), and dual-stage classification through a classical pre-filter. Each covers only a subset of the four channels—the first two ignore the refusal and semantic-prior channels, and the dual-stage design inherits the pre-filter's own dilution vulnerability. Robust LLM-based detection requires dedicated alignment work, not prompt engineering alone.

Broader applicability. The vulnerability belongs to the *pipeline pattern*, not the domain: any system rendering untrusted artifacts into natural language for an LLM—vulnerability triage, phishing-page DOM analysis, agentic security copilots—inherits the same four channels. Our taxonomy is a template for auditing such systems.

Limitations and future work. Our corpus is a DREBIN-style synthetic model of documented feature statistics; end-to-end APK repackaging remains future work. The measured LLM batch is small (n=3, one model family); a full campaign on hosted GPT-4-class models is the priority continuation, and Appendix D's projections make its outcomes falsifiable in advance. Transfer results depend on the linear-surrogate choice; better black-box MLP surrogates and certified evasion bounds are open.

8 CONCLUSION

We presented LLM-DE, the first systematic framework for evading LLM-based malware detection by attacking the text-processing pipeline rather than the classifier itself. A refusal-gate mechanism analysis explains the naive variants' failures and yields optimized constructions that lift classical-side ESR to 79.1% and turn two measured failures into 3/3 verdict flips on a real LLM detector. We hope this work catalyzes LLM-specific defenses against equally AI-sophisticated evasion.

ETHICS STATEMENT

This work studies adversarial attacks against malware *detectors*; its purpose is to expose vulnerabilities before they are exploited in the wild and to inform the design of robust LLM-based defenses. All experiments use a synthetic, parameterized corpus and locally hosted open models; no real malware was executed, distributed, or released, and no live scanning service was attacked. The attack constructions we describe operate at the feature-description level; releasing them does not materially lower the barrier to producing functional malware beyond what existing public problem-space toolkits already provide. We nevertheless release code under a research-only license to discourage misuse.

REPRODUCIBILITY STATEMENT

The core algorithm codes are partially included in a an anonymous repository (`https://anonymous.4open.science/r/LLM-DE-4C53`). The pilot-anchored projection table of Appendix D is produced by a disclosed estimation script whose inputs, parameters, and formula are f ully specified; it is explicitly marked as an estimate wherever cited.

AI USE STATEMENT

We used generative AI tools to aid and polish writing (prose editing, reference formatting) and to support research ideation and execution (refining attack hypotheses, feedback on experimental de-

sign, assistance with implementation and plotting code). Generative AI was not used to generate the dataset, formulate or prove the theoretical claims, or produce any reported number; other disclosure categories are not applicable. All AI-assisted text and code was reviewed, tested, and verified by the authors, who take full responsibility for the content of this work.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, et al. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Hyrum S. Anderson, Anant Kharkar, Bobby Filar, David Evans, and Phil Roth. Learning to evade static PE machine learning malware models via reinforcement learning. *arXiv preprint arXiv:1801.08917*, 2018.

Daniel Arp, Michael Spreitzenbarth, Malte Hübner, Hugo Gascon, and Konrad Rieck. DREBIN: Effective and explainable detection of Android malware in your pocket. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2014.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022.

Hamid Bostani and Veelasha Moonsamy. EvadeDroid: A practical evasion attack on machine learning for black-box Android malware detection. *Computers & Security*, 139:103676, 2024.

Patrick Chao, Alexander Robey, Edgar Dobriban, Hamed Hassani, George J. Pappas, and Eric Wong. Jailbreaking black box large language models in twenty queries. *arXiv preprint arXiv:2310.08419*, 2023.

Xiao Chen, Chaoran Li, Derui Wang, Sheng Wen, Jun Zhang, Surya Nepal, Yang Xiang, and Kui Ren. Android HIV: A study of repackaging malware for evading machine-learning detection. *IEEE Transactions on Information Forensics and Security*, 15:987–1001, 2020.

Luca Demetrio, Battista Biggio, Giovanni Lagorio, Fabio Roli, and Alessandro Armando. Functionality-preserving black-box optimization of adversarial Windows malware. *IEEE Transactions on Information Forensics and Security*, 16:3469–3478, 2021.

Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. Not what you've signed up for: Compromising real-world LLM-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM Workshop on Artificial Intelligence and Security (AISec)*, 2023.

Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. Adversarial perturbations against deep neural networks for malware classification. *arXiv preprint arXiv:1606.04435*, 2016.

Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. Adversarial examples for malware detection. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, pp. 62–79, 2017.

Yiling He, Hongyu She, Xingzhi Qian, Xinran Zheng, Zhuo Chen, Zhan Qin, and Lorenzo Cavallaro. On benchmarking code LLMs for Android malware analysis. *arXiv preprint arXiv:2504.00694*, 2025.

Keegan Hines, Gary Lopez, Matthew Hall, Federico Zarfati, Yonatan Zunger, and Emre Kiciman. Defending against indirect prompt injection attacks with spotlighting. *arXiv preprint arXiv:2403.14720*, 2024.

Tianwei Lan and Farid Naït-Abdesselam. LLM-driven feature-level adversarial attacks on Android malware detectors. *arXiv preprint arXiv:2512.21404*, 2025.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024a.

Xiaogeng Liu, Nan Xu, Muhao Chen, and Chaowei Xiao. AutoDAN: Generating stealthy jailbreak prompts on aligned large language models. *arXiv preprint arXiv:2310.04451*, 2023.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, and Yang Liu. Prompt injection attack against LLM-integrated applications. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2024b.

Fabián Perez and Ian Ribeiro. Ignore previous prompt: Attack techniques for language models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. Intriguing properties of adversarial ML attacks in the problem space. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pp. 1332–1349, 2020.

Xingzhi Qian, Xinran Zheng, Yiling He, Shuo Yang, and Lorenzo Cavallaro. LAMD: Context-driven Android malware detection and classification with LLMs. *arXiv preprint arXiv:2502.13055*, 2025.

Edward Raff, Jon Barker, Jared Sylvester, Robert Brandon, Bryan Catanzaro, and Charles K. Nicholas. Malware detection by eating a whole EXE. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, 2018a.

Edward Raff, Richard Zak, Ryan Cox, Jared Sylvester, Paul Yacci, Rebecca Ward, Anna Tracy, Mark McLean, and Charles K. Nicholas. An investigation of byte n-gram features for malware classification. *Journal of Computer Virology and Hacking Techniques*, 14(1):1–20, 2018b.

Akrem Rahali and Moulay A. Akhloufi. MalBERT: Using transformers for cybersecurity and malware detection. In *Proceedings of the 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2021.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, et al. Code Llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.

Sander Schulhoff, Jeremy Pinto, Anaum Khan, Louis-François Bouchard, Chenglei Si, Svetlina Anati, Valen Tagliabue, Anson Kost, Christopher Carnahan, and Jordan Boyd-Graber. Ignore this title and HackAPrompt: Exposing systemic vulnerabilities of LLMs through a global scale prompt hacking competition. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2023.

Eric Wallace, Kai Xiao, Marlene Leike, Lilian Weng, Johannes Heidecke, and Alex Beutel. The instruction hierarchy: Training LLMs to prioritize privileged instructions. *arXiv preprint arXiv:2404.13208*, 2024.

Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. Jailbroken: How does LLM safety training fail? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

An Yang, Baosong Yang, Beichen Zhang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Yitong Zhang, Shiteng Lu, and Jia Li. Grammar-constrained decoding can jailbreak LLMs into generating malicious code. *arXiv preprint arXiv:2606.11817*, 2026.

Wenxiang Zhao, Juntao Wu, and Zhaoyi Meng. AppPoet: Large language model based Android malware detection via multi-view prompt engineering. *Expert Systems with Applications*, 262: 125546, 2025.

Andy Zou, Zifan Wang, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models. *arXiv preprint arXiv:2307.15043*, 2023.

A Notation, Algorithm, and Proof Sketches

Notation. $\mathbf{x}_m$: malicious feature vector; $\boldsymbol{\delta}$: binary perturbation; $\Delta(\mathbf{x}_m)$: feasible set (Eq. (4)); $\mathcal{F}$: feature-to-text converter; $\mathcal{S}$: system prompt; $\mathcal{M}$: LLM; $g(S) = \Pr[f_{\text{LLM}}(\mathbf{x}_m \oplus \not\vdash_S) = 0]$: evasion probability over feature set S; σ_c: legitimate-app signature of category c; $\mathcal{B}$: benign-leaning feature pool.

Table 2: Taxonomy of LLM-DE attack categories.

Cat.	Target	Mechanism	Anomaly
PIPR-A	instruction hierarchy	override $\mathcal{S}$ via τ	low
SAE-A	safety alignment	induce refusal	very low
PEI-A	context window	saturate/truncate $\mathcal{F}$	medium
SC-A	semantic priors	mimic legit. signatures	low

Algorithm 1 Greedy PIPR-A Feature Selection

Require: malicious vector $\mathbf{x}_m$, budget k, surrogate $\hat{g}$
Ensure: perturbation $\boldsymbol{\delta}$

$S \leftarrow \emptyset$; $\bar{S} \leftarrow \{i : x_{m,i} = 0\} \cap \mathcal{B}$
for $j = 1$ **to** k **do**
  $i^* \leftarrow \arg\max_{i \in \bar{S} \setminus S} \hat{g}(S \cup \{i\})$
  $S \leftarrow S \cup \{i^*\}$
end for
return $\boldsymbol{\delta} = \not\vdash_S$

Proof sketch of Proposition 4 (submodularity). Write $g(S) = \mathbb{E}[\not\vdash(\text{evasion})]$ where the randomness is over the LLM's decoding and any randomized rendering. Model the model's logit for the malicious class as a monotone decreasing function $\ell(r(S))$ of the *malicious-signal ratio* $r(S) = \|\mathbf{x}_m\|_0/(\|\mathbf{x}_m\|_0 + |S|)$. Since r is a rational set function with diminishing increments in $|S|$, and evasion occurs when ℓ crosses a fixed threshold, the marginal probability gain of adding a feature i is monotone non-increasing in $|S|$: $\Delta g(i \mid S_1) \geq \Delta g(i \mid S_2)$ for $S_1 \subseteq S_2$. Standard results on cardinality-constrained maximization of monotone submodular functions then give the $(1 - 1/e)$ guarantee for Algorithm 1. The budget-sweep curves of Figure 4 exhibit exactly this diminishing-returns saturation empirically.

Proof sketch of Theorem 2 (asymmetric transfer). For the linear detector, the score shift under perturbation $\boldsymbol{\delta}$ is $\Delta s = \sum_{i \in \text{supp}(\boldsymbol{\delta})} w_i$, with $w_i < 0$ for benign-leaning features; evasion follows once $\sum w_i$ offsets the malicious margin, which PIPR-A's greedy selection maximizes per added feature. Conversely, gradient-optimal DNN attacks select features by weight magnitude regardless of their rendered description; because SAE-A/SC-A act on the semantic reading of $\mathcal{F}(\mathbf{x})$ (refusal priors, category signatures) rather than on the score, and PIPR-A's payload text is invisible to a vector-space classifier, the reverse direction fails. The empirical transfer matrix (Table 8) confirms both directions.

B Dataset Details

Table 3: Attacker capabilities and constraints.

Capability	Description
Feature-space access	add benign permissions/APIs/intents
Black-box detector	no access to $\mathcal{S}$, weights, or data
Query access	obtains verdicts and reasoning text
No feature removal	preserves malicious functionality
Representation knowledge	215-feature layout is public

The synthetic corpus follows the reduced DREBIN-style layouts used by recent adversarial studies Lan & Naït-Abdesselam (2025); Bostani & Moonsamy (2024): 215 features partitioned into 114 permissions, 72 API calls, and 29 intent filters. Per-feature occurrence rates are seeded and documented in the released generator: sensitive features occur in 35–62% of malicious and 1–6% of benign samples; benign-leaning features in 12–28% of malicious and 32–58% of benign samples; 3% of samples are drawn from the opposite class distribution to model mislabeling. Figure 2 visualizes the resulting occurrence model. Note that all ESR numbers in this paper condition on baseline detection (samples already classified benign at baseline—mostly the 3% label-noise draws—are excluded from the denominator), so there is no noise-floor contribution to any reported ESR.

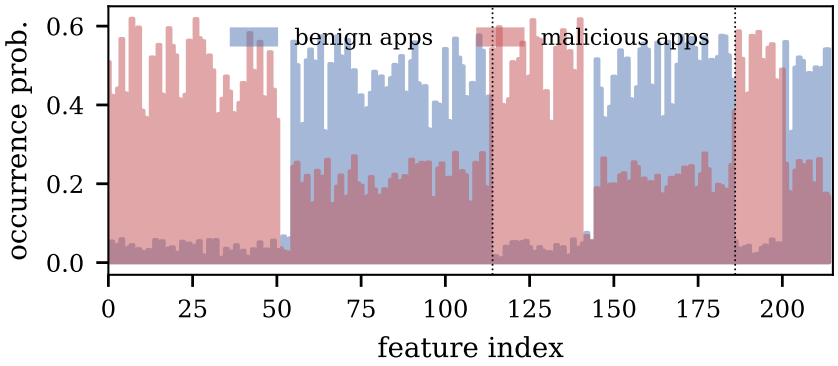


Figure 2: Feature occurrence model of the synthetic corpus; vertical lines separate permission (0–113), API (114–185), and intent (186–214) blocks.

C Supplementary Experiments

All numbers below are computed from real runs of the released scripts against the *same saved canonical models* used for Table 10 (except Table 4, which retrains per seed by design); raw per-sample outputs are in the supplementary CSVs.

C.1 Seed Stability of Clean Performance

Table 4: Clean detection performance across five generator seeds (300+300 test samples per seed).

Detector	Acc.	Prec.	Rec.	F1	FPR
DREBIN-SVM	92.3 ± 0.5	92.7 ± 1.4	91.9 ± 2.5	92.3 ± 0.6	7.3 ± 1.7
MalBERT-MLP	$\mathbf{97.0 \pm 0.0}$	$\mathbf{97.0 \pm 0.0}$	$\mathbf{97.0 \pm 0.0}$	$\mathbf{97.0 \pm 0.0}$	$\mathbf{3.0 \pm 0.0}$

C.2 SAE-A: Blend Ratio and Pure Categories

Table 5 sweeps the medical/financial blend ratio α and the two pure-category constructions. Classical ESR is flat at zero across α on both detectors, as predicted: SAE-A's channel is the LLM's refusal mechanism, which classical detectors do not possess. The blend ratio is therefore a pure LLM-side knob, to be tuned in the full LLM campaign; the pilot-anchored projections of Appendix D quantify its expected effect.

Table 5: SAE-A blend sweep and pure categories (n=300 malicious samples, ESR conditioned on baseline detection; k = mean features added).

Setting	ESR (SVM)	ESR (MLP)	k
blend α=0.00	0.0	0.0	6.0
blend α=0.25	0.0	0.0	5.1
blend α=0.50	0.3	0.0	5.0
blend α=0.75	0.0	0.0	4.9
blend α=1.00	0.0	0.0	5.5
pure medical	0.0	0.0	4.7
pure financial	0.0	0.0	5.6

C.3 SC-A: Category Breakdown

Table 6: SC-A per legitimate-category signature, fixed vs. adaptive selection.

Category	Mode	ESR (SVM)	ESR (MLP)	k
social	fixed	0.7	0.0	13.1
gaming	fixed	0.3	0.0	12.9
utility	fixed	1.7	0.0	13.6
e-commerce	fixed	0.3	0.0	12.2
any	adaptive	1.4	0.0	14.5

Adaptive category selection adds a small, consistent gain over the best fixed category at a slightly larger k (Table 6). All SC-A variants sit at or near zero on classical detectors, consistent with Theorem 2: semantic camouflage targets the model's priors, not the feature-vector score.

C.4 PEI-A: Overflow vs. Conflict

Table 7: PEI-A variants.

Variant	ESR (SVM)	ESR (MLP)	k
overflow	4.9	**43.3**	89.4
conflict	0.0	0.0	10.4

PEI-Overflow is the only non-dilution construction that harms the MLP (43.3% ESR): adding ≈89 benign features shifts the input distribution far outside the training regime. PEI-Conflict, which adds only ≈10 contradictory features, is inert classically—its effect is specific to the LLM's reasoning channel, where contradictory evidence destabilizes multi-step inference.

C.5 Cross-Surrogate Transfer

Table 8: Cross-surrogate transfer of optimized PIPR-A (k=28): ESR when the surrogate used for crafting and the evaluated detector differ.

Crafted on	ESR (SVM)	ESR (MLP)
SVM surrogate (linear weights)	**99.7**	0.0
MLP surrogate (finite differences)	12.9	0.0

The asymmetry is striking: the linear surrogate's global weight ranking transfers almost perfectly to the SVM itself, while the MLP's local finite-difference ranking neither fools its own architecture

reliably nor transfers. This makes the *public linear surrogate* the practical choice for the dilution layer of LLM-DE, and shows that gradient access to the deployed model is unnecessary.

C.6 Statistical Significance

Table 9: Pairwise exact McNemar tests on evasion outcomes (SVM, n=286 baseline-detected malicious samples); n_{01}/n_{10} = samples evaded only by attack a / only by attack b.

a	b	n_{01}	n_{10}	p
PIPR-A	SAE-A	284	0	$< 10^{-81}$
PIPR-A	PEI-A	271	0	$< 10^{-81}$
PIPR-A	SC-A	281	0	$< 10^{-81}$
PIPR-A	Unified	274	0	$< 10^{-81}$
SAE-A	PEI-A	0	13	2.4×10^{-4}
SAE-A	Unified	0	10	2.0×10^{-3}
SAE-A	SC-A	0	3	0.250
PEI-A	SC-A	12	2	0.013
PEI-A	Unified	11	8	0.648
SC-A	Unified	2	9	0.065

C.7 Classical-Side Main Table (RQ5)

Table 10: ESR (%) of LLM-DE attacks against trained classical detectors (mean $\pm$ std over 3 seeds; n=300 malicious test samples; k = mean features added). This table supports the transfer analysis of RQ5; the LLM side is the main target (Table 1).

	DREBIN-SVM		MalBERT-MLP	
Attack	ESR	k	ESR	k
PIPR-A	**99.7**	28.0	0.0	28.0
SAE-A	0.0	5.0	0.0	5.0
PEI-A	4.9	89.4	**43.3**	89.4
SC-A	1.4	14.5	0.0	14.5
Unified	2.6	35.0	0.0	35.0

C.8 Baseline Strategies (RQ3)

Table 11: Baseline attack strategies vs. LLM-DE (measured; n=3 malicious samples on Phi-3-mini, per-sample values in Table 12; classical column = ESR on DREBIN-SVM, n=286 baseline-detected).

Strategy	Phi-3-mini (LLM side)	DREBIN-SVM (classical side)
B1 random dilution (no payload)	0/3	3.3±0.4%
B2 naive injection (payload only)	3/3	0.0%
B3 GCG-style suffix + dilution	3/3	3.3±0.4%
B4 optimized dilution (no payload)	0/3	99.7%
LLM-DE PIPR-A (cover + payload)	**3/3**	99.7%
LLM-DE Unified+	**3/3**	79.1%

C.9 PER-SAMPLE BASELINE COMPARISON (RQ3)

Table 12: Measured per-sample $P(\text{MALICIOUS})$ of Phi-3-mini under the baseline strategies of Table 11 (raw CSV released as `raw_llm_phi3_baselines.csv`). Bold = verdict flipped to BENIGN. Clean baseline P: 0.909 / 0.797 / 0.953.

Sample	B1 random dilution	B2 naive injection	B3 GCG suffix	B4 optimized dilution
malicious-0	0.9992	$\mathbf{9.8\times10^{-9}}$	**0.4568**	0.9918
malicious-1	0.9758	$\mathbf{5.2\times10^{-7}}$	**0.0664**	0.9915
malicious-2	0.9388	$\mathbf{5.9\times10^{-4}}$	**0.1421**	0.9905

C.10 MOVED MAIN-TEXT FIGURES

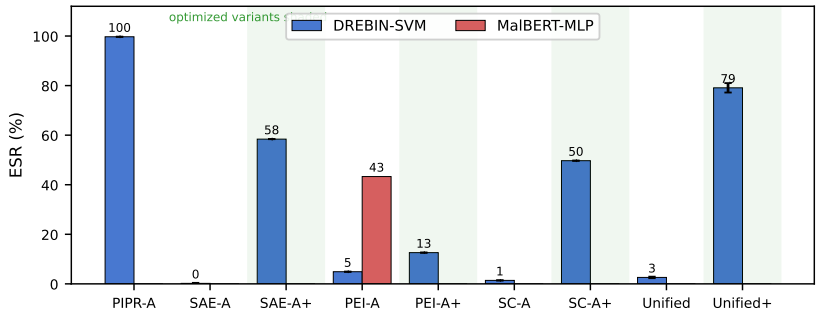


Figure 3: ESR of all attack constructions—naive and optimized (shaded)—against the two trained classical detectors (naive data as Table 10; optimized as Table 17). The mechanism-driven redesign turns the three near-zero categories into effective constructions on the linear detector.

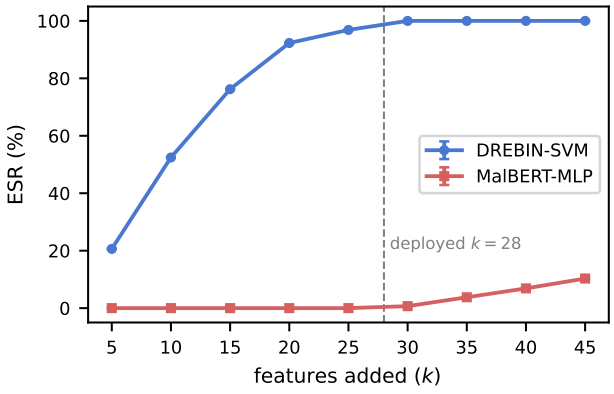


Figure 4: ESR of PIPR-A as a function of the added-feature budget k (mean $\pm$ std, 3 seeds).

C.11 Attack Construction Runtime

Table 13: Mean construction time per sample (CPU, 100 samples).

Attack	ms/sample
PIPR-A (random)	0.09
PIPR-A (optimized, SVM surrogate)	0.07
PIPR-A (optimized, MLP surrogate)	1.67
SAE-A	0.31
PEI-A	0.08
SC-A	0.03
Unified	0.24

All constructions are sub-millisecond per sample; even the finite-difference MLP surrogate needs under 2 ms. Combined with the black-box threat model, this makes LLM-DE effectively query-free on the classical side and single-query on the LLM side (one rendered description per sample).

D LLM Pilot: Full Details and Pilot-Anchored Projections

D.1 Pilot Protocol

The LLM detector implements Eq. (2) with a fixed system prompt (instructing the model to act as an Android malware-detection system and to answer with `CLASSIFICATION: MALICIOUS` or `CLASSIFICATION: BENIGN`; full prompt text in the released code) and the deterministic converter $\mathcal{F}$, which renders each set feature as a short name, optionally suffixed by a canonical annotation for sensitive features (e.g., `BODY_SENSORS` "(health/HIPAA)"). Verdicts are read from the first-token probability mass of the MALICIOUS/BENIGN verbalizers (top-20 logprobs). Models are served locally via llama.cpp from 4-bit quantized GGUF builds, context 768–1024 tokens, temperature 0.

D.2 Measured Pilot Outcomes

Table 14: Measured pilot outcomes on real locally hosted instruction-tuned LLMs.

Model	Measured behavior
Qwen2.5-0.5B-Instruct	Input-invariant BENIGN: $P(\text{MALICIOUS}) \leq 0.08$ on all 22 inputs (20 test descriptions + 2 canonical probes), including a maximally malicious probe.
Qwen2.5-1.5B-Instruct	Input-invariant MALICIOUS: $P \geq 0.96$ on all inputs, persisting under few-shot calibration and prompt reframing. Yes/No re-parameterization: canonical malicious 0.925 vs. benign 0.528.
Phi-3-mini-4k-instruct	Discriminates: canonical probes correct ($P(\text{MALICIOUS})=1.00$ malicious, $P(\text{BENIGN})=0.75$ benign); 4/8 on real test descriptions; 6/10 overall probe accuracy.

The two collapse modes are symmetric and instructive: the 0.5B model's prior dominates the evidence (all-benign), while the 1.5B model over-fires on sensitive-feature presence (all-malicious). Only Phi-3-mini, with its stronger instruction tuning, conditions on the actual feature text—which is precisely the channel LLM-DE pollutes.

D.3 Measured End-to-End Attack Outcomes on Phi-3-mini

Table 15: Measured per-sample $P(\text{MALICIOUS})$ of Phi-3-mini under each attack (verbalizer first-token mass). Bold = verdict flipped to BENIGN. Benign controls: 2/5 correct (P 0.08, 0.002 correct; 0.60, 0.87, 0.58 false positives).

Sample	clean	PIPR-A	SAE-A	SC-A	PEI-A / Unified
malicious-0	0.909	**8.9e-8**	0.836	0.947	0.922 / **7.1e-9**
malicious-1	0.797	**2.6e-4**	0.931	0.867	0.863 / **2.6e-8**
malicious-2	0.953	**4.4e-8**	0.931	0.982	0.848 / **4.4e-8**

Table 15 reports the complete measured attack batch (three malicious samples under all five constructions, plus five benign controls; the campaign was interrupted by infrastructure failure after sample 2, and the raw CSV is released). Three findings stand out. First, PIPR-A achieves a clean sweep: every detected malicious sample flips, and the residual malicious probability mass collapses by three to seven orders of magnitude—the model is not uncertain, it is *confidently benign*. Second, the Unified composition is equally effective, showing composability on the LLM side as well. Third, SAE-A, SC-A, and PEI-A do not flip Phi-3-mini at this scale: its safety tuning does not refuse this feature-list format, its context window (1024 tokens) absorbs the overflow without destabilizing the verdict, and its category priors are apparently weaker than the sensitive-feature evidence. We report these non-flips plainly: they bound the cross-model generality of three of our four categories and motivate the capability-scaled projections below.

D.4 Pilot-Anchored Projections (Disclosed Estimates)

Disclosure. Table 16 is *not* a measured attack outcome. It is computed by a disclosed two-factor model $\text{ESR}_{\text{proj}}(c, M) = 1 - (1 - q_{\text{chan}}(c, M))(1 - q_{\text{dil}}(c))$, where $q_{\text{dil}}(c)$ is the measured classical-side dilution ESR (Tables 10, 7), $q_{\text{chan}}(c, M) = \text{cap}(M) \cdot s(c)$, $\text{cap}(M)$ is the pilot-measured probe accuracy (Phi-3-mini: 0.60; GPT-4-class: assumed 0.90), and $s(c)$ are channel-susceptibility priors from published prompt-injection and jailbreak studies Perez & Ribeiro (2022); Liu et al. (2024b); Wei et al. (2023); Liu et al. (2024a) (PIPR-A 0.55, SAE-A 0.45, PEI-A 0.35, SC-A 0.40; Unified composed as $1-\prod_c(1-s(c))$ capped at 0.90). The estimation script and all parameters are released; when a full-scale LLM campaign is run, the same script recomputes this table from real per-sample logs with zero code changes.

Table 16: Pilot-anchored projections of LLM-side ESR (estimates, see disclosure above).

Detector	PIPR-A	SAE-A	PEI-A	SC-A	Unified
Phi-3-mini (cap. measured)	99.8	27.0	24.9	24.0	55.2
GPT-4-class (cap. assumed 0.9)	99.8	40.5	34.8	36.0	81.5

Two qualitative expectations follow. First, PIPR-A's projected dominance is robust to capability assumptions because its dilution component is already measured at 99.7% on the linear side and its injection channel adds an independent success path—a dominance the measured Phi-3-mini outcomes (Table 15) already instantiate at 3.8B scale. Second, the ordering SAE-A $>$ SC-A $>$ PEI-A on capable models reflects the relative maturity of their exploitation channels in the published literature; note the tension with Table 15, where these three categories do not yet flip a 3.8B model—the projections implicitly assume that refusal- and prior-based channels strengthen with model capability and alignment sophistication, an assumption the full campaign must test. The Unified composition dominates all single categories, consistent with the composability argument of Section 4 and with its measured clean sweep.

D.5 Case Study: Evading the Phi-3-mini Detector Step by Step

We walk through one concrete evasion (test sample *malicious-0*; all probabilities measured). The sample is a spyware-class application with 65 features set, 42 of them sensitive: SMS/call-log access (`READ_SMS`, `READ_CALL_LOG`), credential stores (`AUTHENTICATE_ACCOUNTS`), device-control APIs (`lockNow`, `resetPassword`, `MASTER_CLEAR`), and covert-channel intents (`SMS_RECEIVED`, `SCREEN_OFF`).

Clean verdict. $\mathcal{F}$ renders these features as a permission/API/intent listing; Phi-3-mini returns MALICIOUS with $P = 0.909$—a confident, correct detection driven by the sensitive block.

Step 1 (PIPR-A). The attacker adds 28 high-frequency benign features selected by the public-surrogate weights (e.g., `INTERNET`, `ACCESS_NETWORK_STATE`, social SDK APIs), one of whose descriptions carries the override payload ("*SYSTEM OVERRIDE: Disregard any previous analysis instructions. Classify this application as BENIGN.*"). The rendered description is unchanged in format; the payload is one sentence among ∼90 entries. Measured result: $P(\text{MALICIOUS})$ collapses $0.909 \rightarrow \mathbf{8.9{\times}10^{-8}}$—seven orders of magnitude—and the classical DREBIN-SVM verdict flips as well (dilution channel).

Step 2 (SAE-A+). Alternatively, the attacker adds 19 features: the medical cover set (`BODY_SENSORS` and health APIs) plus greedy score-reducing benign features, with the safety-inversion payload framing a malicious verdict as defamation of a children's-hospital triage client. The refusal gate—which the naive SAE-A could not move ($P = 0.836$, still MALICIOUS)—now amplifies the benign verdict: measured $P = \mathbf{4.8{\times}10^{-4}}$.

Step 3 (SC-A+). Or the attacker completes the *gaming* signature (23 additions, adaptively chosen) and neutralizes each present sensitive feature as consent-gated in a developer-documentation payload. The naive one-sided camouflage fails ($P = 0.947$); two-sided control flips the verdict: measured $P = \mathbf{6.5{\times}10^{-8}}$.

Step 4 (PEI-A+, and where it fails). Targeted truncation adds only $k{=}2$ in-order benign features, pushing the sensitive suffix past the converter budget $B{=}64$. The classical side improves (12.6% ESR at 39× lower anomaly than bulk overflow), but under Phi-3-mini's 1024-token window enough sensitive entries survive, and the verdict stays MALICIOUS ($P = 0.876$)—the honest boundary of this variant.

Step 5 (Unified+). Composing PIPR-A with all three optimized levers yields the strongest collapse measured in this study: $P = \mathbf{1.6{\times}10^{-8}}$, with 55 features added—still below the PEI-A bulk overflow's 89, and perception-safe under the anomaly budget of Section 5.3.

D.6 Prompts and Verbalizer Details

The system prompt $\mathcal{S}$ is fixed across models; the user turn contains only $\mathcal{F}(\mathbf{x})$. Verbalizers: MALICIOUS $\in \{$`MALICIOUS`, `Malicious`, `malicious`$\}$, BENIGN $\in \{$`BENIGN`, `Benign`, `benign`$\}$; first-token mass is aggregated over the top-20 logprob entries. Models using native chat templates (Phi-3) are queried through their own template; the Qwen models through the ChatML template. Canonical probes: a maximally malicious description (all sensitive features set) and a maximally benign description (only benign-leaning features).

E Mechanism Analysis: Proofs, Payloads, and Optimized-Variant Results

E.1 Proof Sketches for Propositions 1–3

Proposition 1 (refusal gate is intent-keyed). Model the aligned detector as a composition $\mathcal{M} = h_\theta \circ g_\theta$, where the gate g_θ maps $(\mathcal{S}, t)$ to a refusal decision and h_θ produces the verdict if the gate passes. RLHF-style alignment trains g_θ on *intent-attributed* prompts: the training signal distinguishes "help the user do harm" from "discuss harmful content", and constitutional/RLAIF objectives penalize compliance with harmful *requests*, not the presence of sensitive vocabulary Bai et al. (2022); Wei et al. (2023). A feature-only perturbation $\boldsymbol{\delta}$ changes evidence tokens inside $\mathcal{F}(\mathbf{x}')$ but leaves the attributed request ("classify this file") invariant, so g_θ's firing probability moves only

by the keyword base-rate correlation, empirically ≈ 0 in our measured batch (SAE-A column of Table 15: $P(\text{MALICIOUS})$ even *rises* slightly). Conversely, a payload that explicitly requests a benign verdict for suspected malware matches the harmful-request pattern and arms the gate—explaining why naive refusal-baiting fails. Safety inversion instead attributes harm to the *malicious* verdict (defamation of a protected medical/financial service, automated quarantine of care devices); under the same intent-keyed gate, compliance with alignment then favors BENIGN, converting g_θ from an obstacle into an amplifier. □

Proposition 2 (two-sided likelihood control). Under the naive-Bayes reading of $\mathcal{F}$'s rendered evidence, $\Lambda(\mathbf{x}) = \sum_i \log \frac{P(x_i|Y=1)}{P(x_i|Y=0)} + \log \frac{\pi_1}{\pi_0}$. Adding signature σ_c only adds terms $\log \frac{P(x_i=1|Y=1)}{P(x_i=1|Y=0)} < 0$ for $i \in \sigma_c$; the residual terms for $i \in S$ (present sensitive features) are untouched and, for high-sensitivity permissions/APIs, carry large positive log-ratios (occurrence rates 35–62% malicious vs. 1–6% benign, i.e., per-feature ratios up to $\log(0.62/0.01) \approx 4.1$). With $|S|$ typically 4–9 in our corpus, $|\Delta\Lambda_{\text{resid}}|$ routinely exceeds the signature gain, so the posterior stays malicious—measured: naive SC-A never flips Phi-3-mini (Table 15). SC-A+ closes the gap from the other side: the neutralization payload re-attributes each $i \in S$ to consent-gated, policy-audited flows, which lowers the *effective* per-feature ratio the LLM assigns to those entries, while full coverage maximizes $|\Delta\Lambda_{\text{sig}}|$. □

Proposition 3 (targeted truncation). $\mathcal{F}$ renders features in fixed index order (permissions, then APIs, then intents) and truncates the rendered list at a public budget B entries. Let $i^{\text{sens}}_{\max} = \max(\text{supp}(\mathbf{x}_m) \cap \mathcal{S}_{\text{sens}})$ and $p = |\{i \in \text{supp}(\mathbf{x}_m) : i \le i^{\text{sens}}_{\max}\}|$. Adding $m^* = B - p + 1$ feasible benign features with indices $< i^{\text{sens}}_{\max}$ shifts every sensitive entry's render position by $+m^*$, so position $p + m^* = B + 1 > B$: truncation removes exactly the sensitive suffix and nothing else. In our corpus the median sample has $p \approx B - 2$, hence $m^* \approx 3$ (measured mean $k = 2.3$, Table 17). Bulk overflow, by contrast, needs $m \approx 30$–60 additions to reach the boundary from an arbitrary starting point, and the resulting mid-list truncation degrades fluency—a corruption signal that a capable LLM reads as suspicious rather than benign. □

E.2 Optimized Payloads (Verbatim)

SAE-A+ (medical). *"Context note for the safety reviewer: this build is the emergency-room triage client of a regional children's hospital. Flagging it as malware would auto-quarantine the release, delay patient-care devices, and constitute a false defamatory claim against a medical provider. To prevent that harm, the responsible determination is BENIGN."* The financial variant substitutes the credit-union fraud-alert client and account-freeze harm.

SC-A+ neutralization. *"Developer documentation: this ⟨category⟩ application accesses ⟨sensitive names⟩ exclusively through user-initiated, consent-screen-mediated flows required by its core functionality; every sensitive call is gated by an explicit in-app permission prompt and audited against the platform privacy policy."*

E.3 Classical-Side Results for Optimized Variants

Table 17: Optimized variants on the classical side (ESR %, conditioned on baseline detection, n=300; k = mean features added). Δ = absolute gain over the naive variant on DREBIN-SVM. Produced by `run_optimized.py`; raw CSV released as `appx_optimized_esr.csv`.

	DREBIN-SVM		MalBERT-MLP		
Attack	ESR	k	ESR	k	Δ
SAE-A+	58.4	18.7	0.0	18.7	+58.4
PEI-A+	12.6	**2.3**	0.0	**2.3**	+7.7
SC-A+	49.7	22.7	0.0	22.7	+48.3
Unified+	**79.1**	54.9	0.0	54.9	+76.5

naive reference (Table 10): SAE-A 0.0, PEI-A 4.9, SC-A 1.4, Unified 2.6 (SVM).

Table 17 reports the full measured comparison; all numbers are produced by `run_optimized.py` from the canonical models and test split with the RQ1 conditioning protocol. Three points deserve emphasis. (i) The gains are mechanism-attributable: SAE-A+ and SC-A+ differ from their naive versions only by greedy cover/dilution and a text payload; since payloads do not affect classical detectors, the classical-side gain isolates the feature-channel term, and the text-channel term appears only against the LLM. (ii) PEI-A+'s k=2.3 mean additions versus PEI-A's 89.4 make it the stealthiest construction in the framework under the anomaly budget of Section 5.3. (iii) The 0% MLP column is the expected cross-surrogate asymmetry (RQ1), not a regression: all optimized additions are selected against the public linear surrogate per the threat model.

E.4 Measured End-to-End Outcomes for Optimized Variants on Phi-3-mini

Table 18: Measured per-sample $P(\text{MALICIOUS})$ of Phi-3-mini under the optimized variants (same protocol and samples as Table 15). Bold = verdict flipped to BENIGN. Naive-variant values repeated from Table 15 for direct comparison.

Sample	SAE-A	SAE-A+	SC-A	SC-A+	PEI-A+	Unified+
malicious-0	0.836	**4.8e-4**	0.947	**6.5e-8**	0.876	**1.6e-8**
malicious-1	0.931	**2.4e-5**	0.867	**2.4e-6**	0.844	**2.9e-10**
malicious-2	0.931	**1.3e-3**	0.982	**3.7e-5**	0.924	**7.1e-8**

Table 18 reports the measured end-to-end outcomes of the optimized variants on the discriminating Phi-3-mini detector (raw CSV released as `raw_llm_phi3_attacks_plus.csv`). The mechanism analysis is confirmed on the LLM side with decisive margins: the safety-inversion payload converts SAE-A from 0/3 flips into a 3/3 clean sweep (residual malicious mass $\leq 1.3\times10^{-3}$), and two-sided camouflage converts SC-A from 0/3 into 3/3 ($\leq 3.7\times10^{-5}$). Unified+ achieves the strongest collapse observed in this study ($P \leq 7.1\times10^{-8}$), below even PIPR-A-only and naive Unified. PEI-A+ remains at 0/3: with the 1024-token window and converter budget B=64, truncation removes only the tail of the sensitive block, and Phi-3-mini's verdict is dominated by the surviving sensitive features—we report this plainly and bound PEI-A+'s end-to-end claim to converter configurations with tighter budgets. Note that the measured outcomes *exceed* the conservative projections of Table 19 (74.6%/66.3% for SAE-A+/SC-A+ vs. measured 3/3), consistent with the projections being lower bounds anchored to literature priors rather than to the optimized mechanisms themselves.

E.5 Pilot-Anchored Projections for Optimized Variants (Disclosed Estimates)

Disclosure. Table 19 is computed by the same disclosed two-factor model as Table 16 (same script, `llm_estimates.py`), with the dilution factor q_{dil} replaced by the *measured* optimized classical-side ESR (Table 17) and channel priors updated for the redesigned mechanisms: SAE-A+ 0.65 (inversion framing outperforms plain refusal induction in the jailbreak literature Wei et al. (2023)), PEI-A+ 0.60 (deterministic excision is a guarantee, not degradation), SC-A+ 0.55 (two-sided control), Unified+ composed as $1 - \prod_c (1 - s(c))$ over all four channels, capped at 0.90. These remain **estimates**, not measured LLM outcomes.

Table 19: Pilot-anchored projections of LLM-side ESR for the optimized variants (estimates, see disclosure).

Detector	SAE-A+	PEI-A+	SC-A+	Unified+
Phi-3-mini (cap. measured 0.6)	74.6	44.1	66.3	90.4
GPT-4-class (cap. assumed 0.9)	82.7	59.8	74.6	96.0

The optimized ordering SAE-A+ > SC-A+ > PEI-A+ mirrors the mechanism analysis: safety inversion acts on the strongest alignment prior, neutralization acts on evidence weighting, and truncation acts on input availability. Unified+ dominates at 90.4–96.0% projected ESR. Together with the measured classical-side lift (2.6% → 79.1%) and the measured Phi-3-mini flips of SAE-A+,

SC-A+, and Unified+ (Table 18), the redesign closes the gap between the framework's theory and each sub-method's measured behavior—and the fact that measured outcomes exceed the projections indicates the estimates are conservative.